Assignment 1:
Reflexive Web Agent with Tools Use

Student ID: 113922002, Name: 許耀文

Github link: yaowen225/CE8014-Assignment1

1. **(10%) Describe your Agentic AI application scenario, including target users, use cases, and problems to be solved.**
   Provide a comprehensive description of your application with at least 2-3 specific use cases, clearly defined target users, and concrete problems to be solved.

   This project develops an automated ESG (Environmental, Social, and Governance) news crawling and summarization system based on the WebVoyager framework.

   Target Users:
   - ESG risk assessment teams in investment institutions
   - ESG analysts and researchers
   - Corporate sustainability department personnel
   - General investors interested in ESG issues

   Use Cases:
   1. Automated ESG News Monitoring
      - Automatically browse ESG news websites
      - Identify and categorize latest news in environmental, social, and governance aspects
      - Generate structured Chinese summary reports

   2. ESG Trend Analysis Support
      - Continuously track ESG-related news for specific companies or industries
      - Analyze news impact on sustainable development
      - Help decision-makers quickly grasp ESG development trends

   3. Cross-language ESG Information Integration
      - Automatically translate English ESG news into Chinese summaries
      - Standardize output format for subsequent analysis
      - Reduce language barriers in information acquisition

   Problems to be Solved:
   1. ESG news information is scattered, and current collection methods need improvement as they are time-consuming and labor-intensive.

2. When considering multiple news sources in the future, significant variations between different news content may make systematic analysis difficult.
3. To transform news into actionable decision-making advice, additional methods need to be considered to ensure proper impact assessment.

2. (10%) Analyze at least 2 potential technical challenges in implementation and propose preliminary solutions.

For each technical challenge, provide detailed analysis including impact assessment and step-by-step solution proposals with feasibility evaluation.

Technical Challenge 1: Web Element Location and Interaction

Impact Assessment:
- WebVoyager framework requires precise element location
- Dynamic loading characteristics of ESG News website
- News content distributed across different page levels

Solutions:
1. Optimize Element Location Strategy
   - Implement digital labeling system
   - Use multi-level selectors
   - Add waiting mechanisms for dynamic loading

2. Error Handling Mechanism
   - Implement error detection in extract_summary.py
   - Provide clear error messages
   - Log interaction process for debugging

Feasibility Assessment:
- Verified implementable through WebVoyager framework
- Error handling mechanism working well
- Maintenance costs currently acceptable

Technical Challenge 2: Multilingual Content Processing and Conversion

Impact Assessment:
- Original news in English
- High difficulty in translating ESG terminology
- Need to maintain output format consistency

Solutions:
1. Prompt Optimization
   - Explicitly require Chinese output in tasks_test.jsonl

- Add translation guidelines in prompts.py
- Standardize summary format specifications

2. Summary Extraction Improvements
- Develop extract_summary.py script
- Implement unified output format
- Automatically generate MD reports

Feasibility Assessment:
- Successfully implemented Chinese summary generation
- Unified and readable format
- Can continuously optimize translation quality

3. **(20%) Explain how your system implements the complete cycle of environment perception, decision making, and action execution.**

Detail the complete workflow of your system, demonstrating how each component interacts within the perception-brain-action cycle.

Based on the WebVoyager framework, our system implements a complete perception-decision-action cycle:

1. Environment Perception
- Webpage State Capture
  * Use WebVoyager's screenshot functionality
  * Identify page element locations
  * Parse DOM structure

- Content Analysis
  * Based on tasks defined in tasks_test.jsonl
  * Identify ESG-related news
  * Determine news category and importance

2. Decision Making
- Task Planning
  * Parse ESG guidelines from prompts.py
  * Select appropriate news articles
  * Determine reading and summary strategy

- Interaction Decisions
  * Determine if page scrolling is needed
  * Select click targets
  * Decide whether to generate summary

3. Action Execution
- Web Operations
  * Execute WebVoyager framework interaction commands
  * Handle page responses
  * Wait for content loading

- Result Output
  * Call extract_summary.py
  * Generate structured summaries
  * Output MD format reports

4. Feedback Processing
- Execution Monitoring
  * Record operation results
  * Monitor execution status
  * Handle exceptional situations
- Result Validation
  * Confirm summary completeness
  * Verify Chinese output
  * Check format specifications

4. (30%) Design and execute 3 test tasks, analyze the results, and propose potential improvements based on the current implementation.

Document the execution of three test cases with comprehensive analysis of results and specific improvement suggestions.

Test Task Design:
Based on tasks_test.jsonl configuration, execute three ESG news crawling tasks:

1. Environmental News Test (ESG News-Environment-1)
Execution Results:
- Successfully retrieved Neste green bond news
- Completed title and date extraction
- Initial execution produced English summaries

Improvements:
- Modified tasks_test.jsonl to add Chinese requirements
- Updated ESG guidelines in prompts.py
- Refined summary format specifications

2. Social News Test (ESG News-Social-1)
Execution Results:

- Successfully retrieved DEI weekly news
- Correctly generated Chinese summaries
- Format met requirements

Improvements:
- Optimize social issue keywords
- Improve news classification logic
- Enhance timeliness judgment

3. Governance News Test (ESG News-Governance-1)
Execution Results:
- Encountered difficulties in news retrieval
- extract_summary.py provided error messages
- Recorded failure reasons

Improvements:
- Expand governance news keywords
- Optimize search strategies
- Add retry mechanisms

Implementation Improvement Focus:

1. Program Structure
- Added extract_summary.py for summary processing
- Created run_and_extract.bat for one-click execution
- Optimized error handling mechanisms

2. User Interface
- Unified MD output format
- Added execution status indicators
- Improved error message display

3. Stability Enhancement
- Improved webpage waiting mechanism
- Enhanced element location strategy
- Strengthened exception handling

4. Future Optimizations
- Support for more news sources
- Implement batch processing functionality
- Add database storage